Large-scale Dictionary Learning for Local Coordinate Coding

Bo Xie[1]
boxie.ntu@gmail.com
Mingli Song[2]
brooksong@ieee.org
Dacheng Tao[1]
dctao@ntu.edu.sg

[1] School of Computer Science
Nanyang Technological University
Singapore 639798

[2] College of Computer Science
Zhejiang University
Hangzhou 310027, China

Abstract

Local coordinate coding has recently been introduced to learning visual feature dictionary and achieved top level performance for object recognition. However, the computational complexity scales linearly with the number of samples, so it does not scale up well for large-scale databases. In this paper, we propose an online learning algorithm which, at every iteration round, only processes one or a mini-batch of random samples (e.g., two hundred samples). Our algorithm theoretically ensures the convergence to the expected objective at infinity. Experiments on object recognition demonstrate the advantage over the original local coordinate coding method in terms of efficiency with comparable performance.

1 Introduction

Dictionary learning is a method to learn dictionary items adapted to data of a given distribution. It is shown that dictionary learned from data is more suited for vision task than universal dictionaries, such as wavelet basis [7]. It plays an important role in many vision tasks, e.g., object categorization and face recognition.

Recently, Local Coordinate Coding (LCC) has shown promising results on learning the local geometry of data points [13]. LCC learns a set of anchor points or dictionary that best reconstruct samples while preserving locality. It finds non-zero coefficients for dictionary items that are neighbors of target samples. It has been successfully applied to learning visual feature dictionary and achieved top level performance for object recognition and related tasks, because visual features are not everywhere on the original high dimensional ambient space but embedded in a low dimensional manifold.

One problem with the original local coordinate coding for learning a visual dictionary is that the time complexity grows linearly with the number of samples. For large-scale datasets of millions of samples, the computational cost becomes unacceptable. For example, it will take several days to learn a 500-item dictionary from a million features. In this paper, we propose an online version that only processes one or a small mini-batch of random samples at every iteration round. This stochastic approach converges almost surely and can handle large-scale datasets.

In Section 2, we review recent methods on visual dictionary learning. In Section 3 we formally introduce local coordinate coding and dictionary learning. Then, we illustrate our online version large-scale algorithm in Section 4. Experiments are given in Section 5 and we conclude in Section 6.

2 Related Work

Various formulations have been proposed to code data points on a high dimension space for nonlinear function learning or classification. Vector Quantization (VQ), or using k-means to learn data cluster centroids, is a simple and popular method in the bag-of-features framework [9]. [8] proposed to use hierarchical k-means to learn a vocabulary tree that is scalable for large dictionary. However, k-means and its variants [14] are best suited for Gaussian distributed clusters while image patches and features are more likely to lie on a manifold. Also, the quantization is limited in its expressive power and incurs high approximation error. Recently, sparse coding is used in visual dictionary learning [12]. Sparse coding enforces less constraint than k-means and achieves lower reconstruction error. To capture manifold geometry of the data distribution, local coordinate coding [13] is proposed and it achieves state-of-the-arts performance on PASCAL VOC 2009 challenge.

In dictionary learning framework, the most popular approach is to alternate between two stages: 1) dictionary update and 2) approximating data with linear combination of dictionary items. This is a batch algorithm - every time, all samples have to be processed to update the dictionary. For sparse coding dictionary learning, K-SVD [1] generalizes k-means clustering to sparse coding objective function. Recently, J. Mairal *et al.* [6] proposed an online learning method for sparse coding which is closely related with ours. While our method uses the online optimization technique in [6], it solves a different objective of local coordinate coding [13].

3 Local Coordinate Coding and Dictionary Learning

In this section, we first introduce local coordinate coding which is suitable for learning on the manifold and discuss its dictionary learning scheme.

3.1 Local Coordinate Coding

In local coordinate coding, we consider the problem of nonlinear function learning on a high dimensional space given a set of sample pairs $(x_1, y_1), \ldots, (x_n, y_n)$, where $x_i \in \mathbb{R}^m$ are the data points and $y_i = f(x_i)$ are the corresponding function values. At first, we learn a set of bases or dictionary $D = [d_1 \ldots d_k] \in \mathbb{R}^{m \times k}$ such that samples can be linearly approximated by their nearby bases. Given the dictionary, local coordinate coding finds a best coding $\alpha(x) \in \mathbb{R}^k$ for a sample x that minimizes the reconstruction error and violation of locality constraint.

In general, we are interested in learning Lipschitz smooth functions defined on a manifold. Although the phenomenon of "curse of dimensionality" exists for high dimensional problems, the approximation upper bound is only proportional to the intrinsic dimensionality of the manifold and is not related to the dimension of ambient space, thus efficient learning can take place.

Definition 1 *[13] (Lipschitz Smoothness) A function $f(x)$ on $\mathbb{R}^m$ is (β,γ)-Lipschitz smooth if $|f(x)-f(x')| \le \beta\|x-x'\|$ and $|f(x)-f(x')-\nabla f(x')^T(x-x')| \le \gamma\|x-x'\|^2$ for any x and x'.*

Suppose $f(x)$ is a (β,γ)-Lipschitz smooth nonlinear function on $\mathbb{R}^m$,the linear approximation error of the function is bounded by the approximation error of the coding $\alpha(x)$, where $\sum_j \alpha(x)^j = 1$ ensures the shift-invariance requirement.

$$\left| f(x) - \sum_j \alpha(x)^j f(d_j) \right| \le \beta\,\|x - D\alpha(x)\| + \gamma \sum_j |\alpha(x)^j|\, \left\|d_j - D\alpha(x)\right\|^2 \tag{1}$$

where $\alpha(x)^j$ is the j-th component of $\alpha(x)$ and d_j is the j-th column of D.

For a practical objective function, note that $D\alpha(x) \approx x$ and there is only a difference of constant by replacing the l_2-norm with its square. Thus, we have

$$\min_{D\in\mathscr{C},\alpha} \frac{1}{2}\|x - D\alpha\|^2 + \mu \sum_j |\alpha^j| \|d_j - x\|^2 \tag{2}$$

where $\mathscr{C} = \{D \| \|d_i\| \le 1, i = 1,\dots,k\}$ is the convex feasible set of D. It is important to constrain the columns of D because we can fix $D\alpha$ and scale D to make $\sum_j |\alpha^j| \|d_j - x\|^2$ arbitrarily small. The first term of the objective function measures reconstruction error and the second term preserves locality in coding.

3.2 Dictionary Learning

Given all the samples, we want to learn a good dictionary that is adapted to the distribution of the samples. An obvious approach is to minimize summed objective function of all data samples over D and α simultaneously, i.e.

$$\min_{D\in\mathscr{C},\alpha_i} \sum_i \left(\frac{1}{2}\|x_i - D\alpha_i\|^2 + \mu \sum_j |\alpha_i^j|\, \left\|d_j - x_i\right\|^2 \right) \tag{3}$$

where x_i is the i-th sample and α_i is its corresponding coding coefficient.

However, the above objective function is not jointly convex over D and α, which makes it difficult to solve. Nevertheless, it is convex over D with fixed α and vice versa. Therefore, we can optimize one variable at a time by fixing another and alternate between the two variables.

3.2.1 Sparse Coding

Specifically, when D is fixed, different α_i can be decoupled into individual sparse coding problems. And they can be further transformed into LASSO/LARS problem [10, 11], where efficient implementation exists. Define a diagonal matrix Λ, whose diagonal elements are $\Lambda_{jj} = \left\|d_j - x\right\|^2$ and $\beta = \Lambda\alpha$. In addition, we assume $d_j \neq x$, thus Λ^{-1} exists. For fixed dictionary D and sample x, optimizing over α can be transformed into optimizing over β:

$$\min_{\beta} \frac{1}{2}\left\|x - D\Lambda^{-1}\beta\right\|^2 + \mu\,\|\beta\|_1 \tag{4}$$

where $\|\beta\|_1 = \sum_j \left|\beta^j\right|$ denotes l_1-norm. After solving β, we can obtain $\alpha = \Lambda^{-1}\beta$.

3.2.2 Dictionary Update

After solving all α, optimizing over D is a constrained quadratic programming problem. To see this, expanding the squares in Eq. 3 and dropping the terms without D leads to

$$\begin{aligned}
D =& \operatorname*{argmin}_{D\in\mathscr{C}} \sum_i \left(\frac{1}{2} \|x_i - D\alpha_i\|^2 + \mu \sum_j |\alpha_i^j| \, \|d_j - x_i\|^2 \right) \\
=& \operatorname*{argmin}_{D\in\mathscr{C}} \sum_i \left(\frac{1}{2} \alpha_i^T D^T D\alpha_i - x_i^T D\alpha_i + \mu \sum_j |\alpha_i^j| \left(d_j^T d_j - 2x_i^T d_j \right) \right) \\
=& \operatorname*{argmin}_{D\in\mathscr{C}} \sum_i \left(\frac{1}{2} \operatorname{tr}\left(D^T D\alpha_i\alpha_i^T\right) - \operatorname{tr}\left(D^T x_i \alpha_i^T\right) + \mu \operatorname{tr}\left(D^T D\Sigma_i\right) - 2\mu \operatorname{tr}\left(D^T x_i \bar{\alpha}_i^T\right) \right) \\
=& \operatorname*{argmin}_{D\in\mathscr{C}} \frac{1}{2} \operatorname{tr}\left[D^T D \left(\sum_i \alpha_i \alpha_i^T + 2\mu\Sigma_i \right) \right] - \operatorname{tr}\left[D^T \left(\sum_i x_i \alpha_i^T + 2\mu x_i \bar{\alpha}_i^T \right) \right]
\end{aligned} \tag{5}$$

where $\bar{\alpha}_i$ is component-wise absolute value of α_i, i.e. $\bar{\alpha}_i^j = \left|\alpha_i^j\right|$ and Σ_i is a diagonal matrix constructed from $\bar{\alpha}_i$.

We can store two matrices $A = \sum_i \alpha_i \alpha_i^T + 2\mu\Sigma_i$ and $B = \sum_i x_i \alpha_i^T + 2\mu x_i \bar{\alpha}_i^T$ and use block-coordinate descent to find the optimal D. In iteration k of dictionary update, we update the j-th column d_j^k when other columns are fixed. Denote a_j and b_j as the j-th columns of matrices A and B, a_{jj} as the (j,j)-th element of A and the dictionary D^k at the iteration k. The updating rule is as follows

$$d_j^{k+1} = \Pi_{\mathscr{C}} \left(d_j^k - \frac{1}{a_{jj}} \left(D^k a_j - b_j \right) \right) \tag{6}$$

where $\Pi_{\mathscr{C}}(\cdot)$ means projection onto the set $\mathscr{C}$.

Another efficient method to update dictionary is stochastic gradient [2]. Instead of optimizing the function Eq. 3, stochastic gradient randomly selects a sample or a small batch of samples at every iteration and update dictionary using projected gradient descent. This method converges in expectation.

4 Online Dictionary Learning

Now we present the online version of dictionary learning with local coordinate coding. The key ingredient is to randomly select data points in each iteration round and optimize dictionary using cumulated history information and the algorithm can converge to expectation at infinity.

In general, our ideal objective function is the expectation of local coordinate coding loss function over the sample distribution. However, it is difficult to obtain and it is costly using empirical mean to approximate the expectation in every iteration round. Therefore, we use its upper bound which only requires a random sample per iteration. With this formulation, the online algorithm can scale up to millions of data gracefully.

The ideal objective function for a single sample is as follows

$$f(D,x) = \min_{\alpha} \frac{1}{2} \|x - D\alpha\|^2 + \mu \sum_j |\alpha^j| \, \|d_j - x\|^2 \tag{7}$$

Algorithm 1 Online Dictionary Learning for LCC

```
input: $D_0$ (initial dictionary), $\mu$ and $T$ (number of iterations)
initialize: $A_0 \leftarrow 0, B_0 \leftarrow 0$
for $t \leftarrow 1$ to $T$ do
    Draw a random sample $x_t$ from $p(x)$.
    Local coordinate coding: compute using Eqn. 4
        $\alpha_t = \underset{\alpha}{\operatorname{argmin}} \frac{1}{2}\|x_t - D_{t-1}\alpha\|^2 + \mu \sum_j |\alpha^j| \left\|(D_{t-1})_j - x_t\right\|^2$
    Update $A_t \leftarrow A_{t-1} + \alpha_t \alpha_t^T + 2\mu\Sigma_t$.
    Update $B_t \leftarrow B_{t-1} + x_t \alpha_t^T + 2\mu x_t \bar{\alpha}_t^T$.
    Update dictionary using Alg. 2, such that
        $D_t = \underset{D \in \mathscr{C}}{\operatorname{argmin}} \frac{1}{2} \operatorname{tr}\left(D^T D A_t\right) - \operatorname{tr}\left(D^T B_t\right)$
end for
return $D_T$
```

We also define its empirical mean over n samples

$$f_n(D) = \frac{1}{n}\sum_{i=1}^{n} f(D, x_i) \tag{8}$$

Note that it is generally not convex in D because of the min operator on α.

The optimal dictionary is

$$\hat{D} = \underset{D \in \mathscr{C}}{\operatorname{argmin}}\ \mathbb{E}_x\left[f(D,x)\right] = \underset{D \in \mathscr{C}}{\operatorname{argmin}} \lim_{n \to \infty} f_n(D) \tag{9}$$

To enable online learning, we propose an upper bound for Eq. 8 that accumulates history information and only processes one data point at a time. Suppose at iteration t, we randomly draw a sample x_t from the distribution $p(x)$ and now have a sequence of random samples $x_1, \ldots, x_t$. The objective function at iteration t is defined as

$$\tilde{f}_t(D) = \frac{1}{t}\sum_{i=1}^{t}\left(\frac{1}{2}\|x_i - D\alpha_i\|^2 + \mu \sum_j |\alpha_i^j| \left\|d_j - x_i\right\|^2\right) \tag{10}$$

And

$$\alpha_i = \underset{\alpha}{\operatorname{argmin}} \frac{1}{2}\|x_i - D_{i-1}\alpha\|^2 + \mu \sum_j |\alpha^j| \left\|(D_{i-1})_j - x_i\right\|^2 \tag{11}$$

where $(D_{i-1})_j$ denotes the j-th column of D_{i-1}. Note that α_t is computed from D_{t-1} and x_t and thus decouples from future dictionary values D_k, $(k \geq t)$.

The dictionary update objective is thus

$$D_t = \underset{D \in \mathscr{C}}{\operatorname{argmin}} \tilde{f}_t(D) \tag{12}$$

Algorithm 2 Dictionary Update by Coordinate Descent

```
input: $D = [d_1, \dots, d_k] \in \mathbb{R}^{m \times k}$, $A = [a_1, \dots, a_k] \in \mathbb{R}^{k \times k}$, $B = [b_1, \dots, b_k] \in \mathbb{R}^{m \times k}$.
repeat
  for $j \leftarrow 1$ to $k$ do
    Update $d_j$
        $$u_j \leftarrow d_j - \frac{1}{a_{jj}}(Da_j - b_j)$$
        $$d_j \leftarrow \frac{u_j}{\max(1, \|u_j\|)}$$
  end for
until convergence
return $D$
```

It is easy to rewrite the expression into a quadratic programming problem as in Eq. 5 and the problem can be solved using block-coordinate descent similarly.

Note that we do not need to store all the previous x_i and α_i but only two cumulated matrices $A = \sum_i \alpha_i \alpha_i^T + 2\mu\Sigma_i$ and $B = \sum_i x_i \alpha_i^T + 2\mu x_i \bar{\alpha}_i^T$.

Detailed algorithm for dictionary learning is given in Alg. 1 and coordinate descent dictionary update is given in Alg. 2.

We can prove that $\tilde{f}_t(D_t) - \mathbb{E}_x[f(D_t, x)]$ converges almost surely to 0.

Proof Sketch

First, we will prove that the positive sequence $u_t = \tilde{f}_t(D_t) \geq 0$ is a quasi-martingale by showing that the expected sum of positive jumps $\mathbb{E}[\mathbb{E}[u_{t+1} - u_t | P_t]^+]$ is bounded, where P_t denotes past information at iteration t and $[\cdot]^+$ denotes the positive part of a number.

Then, from this result we have $\sum_{i=1}^{\infty} \frac{\tilde{f}_t(D_t) - f_t(D_t)}{t+1}$ is bounded. We can go on to show that almost surely

$$\tilde{f}_t(D_t) - f_t(D_t) \xrightarrow[t \to \infty]{} 0 \tag{13}$$

5 Experiments

We conducted two experiments on object recognition to compare with the batch version. From the experiments, we can see that our online version converges faster than the batch counterpart and it achieves comparable performance on the final classification results.

5.1 Pascal VOC 2007

PASCAL VOC 2007 [4] contains 5011 images for training and 4952 for test, with a total of 20 categories. We extracted dense SIFT features with sizes 16×16, 24×24 and 32×32. And we used 1 million of these SIFT features to learn the dictionary.

The number of dictionary items k was set to 500. And the trade-off coefficient μ for both algorithms was set to 0.15 to yield around 10 non-zero elements. Our algorithm used a random mini-batch size of 256 at every iteration and iterated for 10,000 times while the batch algorithm iterated for 15 times.

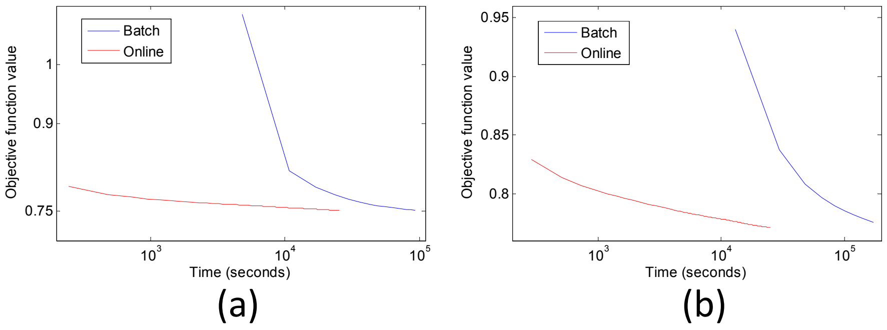

Figure 1: Objective value v.s. time (a) VOC dataset; (b) Caltech 256 dataset.

Table 1: Averaged precision results of 20 categories for batch and online algorithms.

	aero	bicyc	bird	boat	bottle	bus	car	cat	chair	cow
Batch	54.5	32.0	23.0	46.3	10.8	28.9	51.7	39.2	34.0	21.2
Online	56.6	29.7	25.7	50.2	9.53	31.6	53.0	38.4	32.5	25.3
	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Batch	28.4	27.3	53.6	33.8	61.7	14.7	34.2	33.0	56.4	32.0
Online	20.6	23.0	57.1	33.3	60.4	11.0	24.9	23.7	51.0	32.4

After learning the dictionary, we calculated the coding coefficients for SIFT features and used spatial pyramid matching [12] with max pooling. Our spatial pyramid arrangement was 1×1, 2×2 and 3×1. The 8×500 dimensional image feature vector was fed into linear SVM [3] for final classification, where we used 5-fold cross validation to determine the best model parameter.

In Fig. 1 (a), we show the optimization process of batch and online algorithms. The objective function value was evaluated at a separate test set of size 2000 after every iteration round and the learning time is plotted on a logarithmic scale. The time was measured when each algorithm ran on eight cores of 2.8GHz CPU. We can see that each round of the batch algorithm took a long time while our online version converged much faster and arrived at similar objective values orders of time earlier.

Finally, we show the averaged precision results of 20 categories for both algorithms in Table 1. From these results, we can see that the performance of batch and online algorithms is comparable – for some categories, batch version achieved higher precision while for other categories, online method performed better.

5.2 Caltech 256

There are a total of 29,780 images in Caltech 256 [5] with 256 different classes. 1 million random SIFT features with sizes 16×16, 24×24 and 32×32 were used to learn the dictionary of size 500. The trade-off parameter μ for both algorithms was also set to 0.15. The mini-batch size of the online algorithm was 128. The online algorithm iterated for 10,000 times and the batch counterpart iterated for 10 times.

The objective values v.s. time of both algorithms are plotted in Fig. 1 (b). We can see

Table 2: Averaged accuracies for batch and online algorithm on Caltech 256 dataset.

# of training images	45	60
Batch	36.31	38.94
Online	36.42	39.07

that the first iteration of batch algorithm took more than 10,000 seconds (the corresponding time of the left end of the blue curve). By then the online algorithm had already achieved far lower objective function value. The batch algorithm achieved similar objective function value only after 100,000 seconds. From this, it is clear that our online algorithm can scale up to millions of data points gracefully.

We used similar setting as that in the last experiment to construct image feature vectors from SIFT coding coefficients. And we used two kinds of training/test split: in one scheme, 45 images from each category were randomly selected as training set and the rest constituted test set; in the other scheme, 60 images from each category were used for training and the rest for test. Each scheme was repeated for five times and the averaged results are reported in Table 2. Compared with batch algorithm, our online learning scheme achieves similar results.

6 Conclusion

We proposed an online dictionary learning algorithm for local coordinate coding. By randomly selecting samples at each iteration round and accumulating history information, our algorithm can scale up gracefully to millions of samples and is guaranteed to converge almost surely. We demonstrated the performance speed-up over the batch version in our object recognition experiments.

In future work, we will investigate how to choose the number of iterations for a specific training size. Also, how to incorporate discriminative information in dictionary learning is also an important issue.

Acknowledgements

This work was supported by NTU NAP Grant with project number M58020010, the Open Project Program of the State Key Lab of CAD&CG (Grant No. A1006), Zhejiang University, the National Natural Science Foundation of China under Grant 60873124, the Natural Science Foundation of Zhejiang Province under Grant Y1090516, and the Fundamental Research Funds for the Central Universities under Grant 2009QNA5015.

References

[1] Michal Aharon, Michael Elad, and Alfred Bruckstein. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, 2006.

[2] Leon Bottou and Olivier Bousquet. The tradeoffs of large scale learning. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 161–168. MIT Press, Cambridge, MA, 2008.

[3] Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[4] Mark Everingham, Luc Van Gool, Chris Williams, and Andrew Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. URL `http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html`.

[5] Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007. URL `http://authors.library.caltech.edu/7694`.

[6] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online dictionary learning for sparse coding. In *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning*, pages 689–696, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-516-1.

[7] Joseph F. Murray and Kenneth Kreutz-Delgado. Learning sparse overcomplete codes for images. *J. VLSI Signal Process. Syst.*, 46(1):1–13, 2007. ISSN 0922-5773.

[8] David Nister and Henrik Stewenius. Scalable recognition with a vocabulary tree. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2161–2168, Washington, DC, USA, 2006. IEEE Computer Society. ISBN 0-7695-2597-0.

[9] Eric Nowak, Frédéric Jurie, and Bill Triggs. Sampling strategies for bag-of-features image classification. In *European Conference on Computer Vision*. Springer, 2006. URL `http://lear.inrialpes.fr/pubs/2006/NJT06`.

[10] Karl Sjöstrand. Matlab implementation of LASSO, LARS, the elastic net and SPCA, jun 2005. URL `http://www2.imm.dtu.dk/pubdb/p.php?3897`. Version 2.0.

[11] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society (Series B)*, 58:267–288, 1996.

[12] Jianchao Yang, Kai Yu, Yihong Gong, and Thomas S. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *CVPR*, pages 1794–1801. IEEE, 2009.

[13] Kai Yu, Tong Zhang, and Yihong Gong. Nonlinear learning using local coordinate coding. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 2223–2231. 2009.

[14] Tianyi Zhou and Dacheng Tao. Fast gradient clustering. *NIPS 2009 Workshop on Discrete Optimization in Machine Learning: Submodularity, Sparsity & Polyhedra (NIPS: DISCML)*, pages 1–6, 2009.